Coding Arduino

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
#define button 7
byte speaker = 8;
int bel;
void setup() {
  pinMode(button, INPUT_PULLUP);
  pinMode(speaker, OUTPUT);
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();
  bel = 0;
  lcd.print("Smart Bel!");
  delay(3000);
  lcd.setCursor(0, 0); lcd.print("Selamat Datang");
  lcd.setCursor(0, 1); lcd.print("Tekan Bel");
}

void loop() {
  int btnClick = digitalRead(button);
  if (btnClick == LOW && bel == 0) {
    lcd.clear();
    lcd.setCursor(0, 0); lcd.print("Mohon tunggu");
    lcd.setCursor(0, 1); lcd.print("Sebentar ...");
    nada();
  }
  else {
    tone(speaker, 0);
  }
}
void nada() {
  tone(speaker, 262); delay(500);
  tone(speaker, 294); delay(500);
  tone(speaker, 330); delay(500);
  tone(speaker, 349); delay(500);
  tone(speaker, 395); delay(500);
  tone(speaker, 440); delay(500);
  tone(speaker, 494); delay(500);
  tone(speaker, 523); delay(500);
}
```

```
================================================
                Configurasi \input Output
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
#define button 7
byte speaker = 8;
int bel;
void setup() {
  pinMode(button, INPUT_PULLUP);
  pinMode(speaker, OUTPUT);
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();
  bel = 0;
================================================
                        LCD

  lcd.print("Smart Bel!");
  delay(3000);
  lcd.setCursor(0, 0); lcd.print("Selamat Datang");
  lcd.setCursor(0, 1); lcd.print("Tekan Bel");
}

void loop() {
  int btnClick = digitalRead(button);
  if (btnClick == LOW && bel == 0) {
    lcd.clear();
    lcd.setCursor(0, 0); lcd.print("Mohon tunggu");
    lcd.setCursor(0, 1); lcd.print("Sebentar ...");
```

```
          BEL

  nada();
  }
  else {
    tone(speaker, 0);
  }
}
void nada() {
```

```
  tone(speaker, 262); delay(500);
  tone(speaker, 294); delay(500);
  tone(speaker, 330); delay(500);
  tone(speaker, 349); delay(500);
  tone(speaker, 395); delay(500);
  tone(speaker, 440); delay(500);
  tone(speaker, 494); delay(500);
  tone(speaker, 523); delay(500);
}
```

Coding Wemos

```
#include <Adafruit_VC0706.h>
#include <SPI.h>
#include <SD.h>
#include <SoftwareSerial.h>
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>

#define min(a,b) ((a)<(b)?(a):(b))

const char* ssid = "Fitri"; //Nama Wifi di ubah
const char* password = "bismilahbae"; //password di ubah

const char* host = "api.telegram.org";
const int httpsPort = 443;

const char* token = "852306725:AAF3wMbwaHhs_QlnyXPQA2pTo1f1xJ8iCzo"; //
BOT TOken perlu diganti sesuai BOT
const char* chat_id = "784189260"; // Perlu diganti

const char* boundry = "<delimitador_conteudo>";

WiFiClientSecure client;
UniversalTelegramBot bot(token, client);

int Bot_mtbs = 1000;
long Bot_lasttime;

#define chipSelect D10
```

```
#define tombol D8
SoftwareSerial cameraconnection = SoftwareSerial(D14, D15);
Adafruit_VC0706 cam = Adafruit_VC0706(&cameraconnection);


void setup()
{
  pinMode(D0, OUTPUT);
  pinMode(tombol, INPUT_PULLUP);
  Serial.begin(9600);
  Serial.println("VC0706 Camera snapshot test");

  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
    return;
  }
  else {
    Serial.println("MASUK");
  }
  initWifiConnection();
}

void loop()
{
  if (millis() > Bot_lasttime + Bot_mtbs)  {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

    while (numNewMessages) {
      Serial.println("got response");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }

    Bot_lasttime = millis();
  }

  else if (digitalRead(tombol) == LOW) {
    capture();
  }
}

void capture() {
```

```
if (cam.begin()) {
  Serial.println("Camera Found:");
} else {
  Serial.println("No camera found?");
  return;
}
char *reply = cam.getVersion();
if (reply == 0) {
  Serial.print("Failed to get version");
} else {
  Serial.println("-----------------");
  Serial.print(reply);
  Serial.println("-----------------");
}

//cam.setImageSize(VC0706_640x480);        // biggest
//cam.setImageSize(VC0706_320x240);        // medium
cam.setImageSize(VC0706_160x120);          // small

uint8_t imgsize = cam.getImageSize();
Serial.print("Image size: ");
if (imgsize == VC0706_640x480) Serial.println("640x480");
if (imgsize == VC0706_320x240) Serial.println("320x240");
if (imgsize == VC0706_160x120) Serial.println("160x120");

Serial.println("Snap in 3 secs...");
delay(4000);

if (! cam.takePicture())
  Serial.println("Failed to snap!");
else
  Serial.println("Picture taken!");

char filename[13];
strcpy(filename, "IMAGE00.JPG");
SD.remove(filename);
for (int i = 0; i < 100; i++) {
  filename[5] = '0' + i / 10;
  filename[6] = '0' + i % 10;
  if (! SD.exists(filename)) {
    break;
  }
}
```

```
  Serial.print("Saving ");
  Serial.println(filename);

  File imgFile = SD.open(filename, FILE_WRITE);
  uint16_t jpglen = cam.frameLength();
  Serial.print("Storing ");
  Serial.print(jpglen, DEC);
  Serial.print(" byte image.");

  int32_t time = millis();

  byte wCount = 0;
  while (jpglen > 0) {
    uint8_t *buffer;
    uint8_t bytesToRead = min(32, jpglen);
    buffer = cam.readPicture(bytesToRead);
    imgFile.write(buffer, bytesToRead);

    if (++wCount >= 64) {
      Serial.print('.');
      wCount = 0;
    }

    jpglen -= bytesToRead;
    delay(20);
  }
  imgFile.close();

  time = millis() - time;

  Serial.println("done!");
  // Serial.print(time); Serial.println(" ms elapsed");
  sendPhotoToTelegram(filename);
}

void initWifiConnection()
{
  Serial.println();
  Serial.printf("Connecting to Wifi [%s]...\r\n", ssid);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
```

```
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi Connected");
}

void receiveDataFromTelegram()
{
  // Espera por tempo de resposta do Servidor
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println("Response From Telegram!");
      client.stop();
      return;
    }
  }

  Serial.println();
  Serial.println("Receiving from telegram...");

  int responseContentLength = 0;
  while (client.available()) {

    String line = client.readStringUntil('\r');
    client.read(); // lê o caracter '\n'

    Serial.println(line);

    if (line.startsWith("Content-Length:")) {
      int index = line.indexOf(':');
      responseContentLength = line.substring(index + 1).toInt();
    }

    if (line.length() == 0)
      break;
  }

  while (responseContentLength > 0)
  {
    char ch = client.read();
    Serial.print(ch);
    responseContentLength--;
```

```
  }

  Serial.println();
  Serial.println("closing connection");
}

void handleNewMessages(int numNewMessages) {
  Serial.println("handleNewMessages");
  Serial.println(String(numNewMessages));

  for (int i = 0; i < numNewMessages; i++) {
    String chat_id = String(bot.messages[i].chat_id);
    String chat_id_1 = String(bot.messages[i].chat_id);
    String text = bot.messages[i].text;
    String from_name = bot.messages[i].from_name;
    if (text == "/start") {
      String welcome = " Halo " + from_name + "! Alat Pendeteksi Kehadiran Tamu
siap digunakan. ";
      welcome += "Bot akan mengirim gambar otomatis ketika bel ditekan.\n";
      welcome += "Atau ketik /jepret untuk mengambil gambar secara manual\n";
      bot.sendChatAction(chat_id, "typing");
      bot.sendMessage(chat_id, welcome);
    }

    if (text == "/jepret") { //instruksi untuk memanggil foto
      bot.sendChatAction(chat_id, "typing");
      capture();
    }
  }
}

void sendPhotoToTelegram(String filename)
{
  Serial.printf("Connecting Telegram %s:%d... ", host, httpsPort);

  if (!client.connect(host, httpsPort)) {
    Serial.println("Failde To Conenct!");
    return;
  }
  sendDataToTelegram(filename);
  receiveDataFromTelegram();
}
```

```
void sendDataToTelegram(String file_name)
{
  String start_request = "";
  String end_request = "";

  //    String chat_id = String(bot.messages[0].chat_id);
  //    String chat_id_1 = String(bot.messages[0].chat_id);

  start_request = start_request + "--" + boundry + "\r\n";
  start_request = start_request + "content-disposition: form-data; name=\"chat_id\"" +
"\r\n";
  start_request = start_request + "\r\n";
  start_request = start_request + chat_id + "\r\n";

  start_request = start_request + "--" + boundry + "\r\n";
  start_request = start_request + "content-disposition: form-data; name=\"photo\";
filename=\"foto.jpg\"\r\n";
  start_request = start_request + "Content-Type: image/jpeg\r\n";
  start_request = start_request + "\r\n";

  end_request = end_request + "\r\n";
  end_request = end_request + "--" + boundry + "--" + "\r\n";

  //  String file_name = "IMAGE00.jpg";

  Serial.print("Sending ");
  Serial.println(file_name);

  File file = SD.open(file_name);
  int contentLength = (int)file.size() + start_request.length() + end_request.length();

  String headers = String("POST /bot") + token + "/sendPhoto HTTP/1.1\r\n";
  headers = headers + "Host: " + host + "\r\n";
  headers = headers + "User-Agent: ESP8266" + String(ESP.getChipId()) + "\r\n";
  headers = headers + "Accept: */*\r\n";
  headers = headers + "Content-Type: multipart/form-data; boundary=" + boundry +
"\r\n";
  headers = headers + "Content-Length: " + contentLength + "\r\n";
  headers = headers + "\r\n";
  headers = headers + "\r\n";

  Serial.println();
  Serial.println("Mengirim data ke Telegram ...");
```

```
  Serial.print(headers);
  client.print(headers);
  client.flush();

  Serial.print(start_request);
  client.print(start_request);
  client.flush();

  Serial.println("sendFile");
  sendFile(&file);
  file.close();
  client.flush();

  Serial.print(end_request);
  client.print(end_request);
  client.flush();
}

void sendFile(Stream* stream)
{
  size_t bytesReaded;
  size_t bytesSent;
  do {
    uint8_t buff[1024];
    bytesSent = 0;
    bytesReaded = stream->readBytes(buff, sizeof(buff));
    if (bytesReaded) {
      bytesSent = client.write(buff, bytesReaded);
      client.flush();
    }
  } while ( (bytesSent == bytesReaded) && (bytesSent > 0) );
}
```